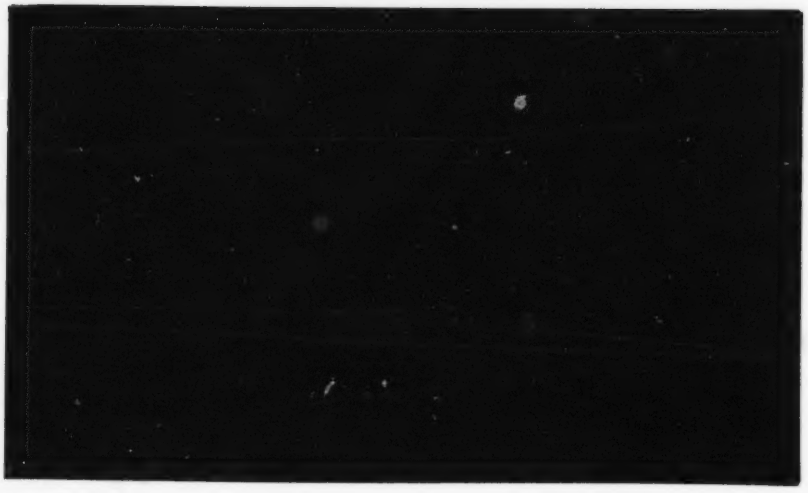
CENTRE
DE RECHERCHE
SUR LES TRANSPORTS
(CRT)
CENTRE
FOR RESEARCH
ON TRANSPORTATION

UNIVERSITÉ DE MONTRÉAL
ÉCOLE DES HAUTES ÉTUDES COMMERCIALES
ÉCOLE POLYTECHNIQUE



C.P. 6128, SUCCURSALE A
MONTRÉAL (QUÉBEC)
CANADA
H3C 3J7

TÉLÉPHONE : (514) 343-7575
TÉLEX : 05-24146 BIBPOLYTEC
TÉLÉCOPIEUR (FAX) : (514) 343-7121

CRT-644

CAPACITATED VEHICLE ROUTING ON TREES

by

Martine Labbé*,
Gilbert Laporte**,
and
Hélène Mercure**

* Econometrisch Instituut, Erasmus Universiteit Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, The Netherlands.

** Centre de recherche sur les transports, Université de Montréal, C.P. 6128, Succursale A, Montréal (Québec) CANADA H3C 3J7.

Université de Montréal
Centre de recherche sur les transports - Publication #644
July 1989

89-05956/

Dépôt légal - 3e trimestre 1989

ABSTRACT

Let T=(V,E) be a tree with nonnegative weights q_k associated with each of its n vertices v_k. A fleet of vehicles of capacity Q is located at the depot represented by vertex v_1. The Capacitated Vehicle Routing Problem on Trees (TCVRP) consists of determining vehicle collection routes starting and ending at the depot such that (i) the weight associated with any given vertex is collected by exactly one vehicle; (ii) the sum of all weights collected by a vehicle does not exceed Q; (iii) a linear combination of the number of vehicles used and of the total distance travelled by these vehicles is minimized. The TCVRP is shown to be NP-hard. This paper presents lower bounds for the TCVRP based on the solutions of associated bin packing problems. A linear time heuristic (upper bound) procedure is then developed and a bound on its worst case performance is presented. These lower and upper bounds are then embedded in an enumerative algorithm. Numerical results follow.

Key words : routing problems, trees, bin packing.

- - - - - -

RÉSUMÉ

Le problème de tournées de véhicules avec capacité sur un arbre T (TCVRP) consiste à déterminer des tournées pour un ensemble de véhicules basés en un dépôt situé au sommet v_1 de T=(V,E), telles que (i) le poids q_k associé à tout sommet v_k de T soit collecté par exactement un véhicule; (ii) la somme de tous les poids collectés par un véhicule ne dépasse pas la capacité Q, et; (iii) une combinaison linéaire du nombre de véhicules utilisés et de la distance totale parcourue par ces véhicules soit minimum. On démontre que le TCVRP est un problème NP-difficile. Cet article présente, pour ce problème, des bornes inférieures basées sur les solutions de problèmes de "bin-packing" associés. Une heuristique (borne supérieure) de complexité linéaire est ensuite développée et une borne sur sa performance dans le pire des cas est obtenue. Finalement, ces bornes inférieures et supérieures sont intégrées dans un algorithme énumératif. Des résultats numériques complètent l'article.

Mots-clés : problèmes de tournées, arbres, bin packing.

The capacitated vehicle routing problem (CVRP) can be defined as follows. Let $G=(V,E)$ be a connected graph where $V=\{v_1, \ldots, v_n\}$ is the vertex set and v_1 denotes the vertex at which the depot is located; $E=\{(v_i, v_j)\}$ is the arc set. Let ℓ_{ij} be the length of arc (v_i, v_j) and $q_i > 0$ be an integer weight associated with v_i ($q_1 = 0$). There is a fleet of homogeneous vehicles of capacity Q based at the depot ($Q \geq q_i$, $i=1, \ldots, n$). The classical CVRP consists of determining vehicle collection routes starting and ending at the depot such that

(i) the weight associated with any given vertex is collected by exactly one vehicle;

(ii) the sum of all weights collected by a vehicle does not exceed Q;

(iii) a linear combination of the number of vehicles used and of the total distance travelled by these vehicles is minimized.

The CVRP and related problems have been extensively studied (see, for example, Laporte and Nobert 1987 and Golden and Assad 1988). The CVRP is NP-hard, even if the assumption that weights cannot be split among several vehicles is relaxed. However, when the sum of the three smallest weights exceeds Q, the problem can be solved in polynomial time by means of a generalized minimum cost matching algorithm, since no vehicle can receive more than two weights (Christofides 1985). Several lower bounds and heuristics have been developed for the CVRP and worst case bounds were obtained for a number of heuristics (Altinkemer and Gavish 1985, 1987 and Haimovich et al. 1988). These were derived in contexts where inter-vertex distances satisfy the triangle inequality; they also presuppose the availability of an α-optimal travelling salesman tour on G. The worst case ratios are $1 + (1-1/Q)\alpha$ in the case where all weights q_i are identical (Altinkemer and Gavish 1985) and $2 + (1-2/Q)\alpha$ in the case of unequal weights (Altinkemer and Gavish 1987). In the current state of knowledge, the smallest value of α which can be determined in polynomial time is equal to 3/2 (see Christofides 1989).

In this paper, we consider a special case of the CVRP where G is a tree. There are a number of practical situations where tree structures are encountered. River networks and some railway networks are such examples. Tree networks are also found in pit mine railways and in newly developed mining or logging areas (e.g. in Northern Canada and in the Amazon) where road construction costs far exceed routing costs.

It is shown that the problem (the TCVRP) is still NP-hard. It is however possible to solve to optimality much larger problems on trees than on general graphs, by properly exploiting their structure. We use the notation already introduced, except that G=T=(V,E) is assumed to be an undirected tree rooted at v_1 so that (v_i, v_j) now represents an edge and by convention, v_i is closer to v_1 than v_j is. In addition, define t_j as the length of the (shortest) path between v_j and v_1.

As we just mentioned, the TCVRP is a NP-hard problem. In the special case where the objective consists of minimizing the number of vehicles, the TCVRP reduces to a bin packing problem (this is true, even in the case of general graphs). Vehicle fixed costs f can easily be taken into account by adding an extra vertex v_0 with $q_0=0$ and $\ell_{0,1}=f$, and by relocating the depot at that vertex. Our results will therefore be developed for the case where the objective is to minimize z, the total distance travelled. Note that with such an objective, the number of vehicles in the solution is not necessarily minimized. Consider indeed the tree depicted in figure 1 and let Q=10. Using two vehicles with associated vertex sets $\{v_3, v_5, v_6\}$ and $\{v_2, v_4\}$ yields a total distance of 14, whereas using three vehicles with associated vertex sets $\{v_2\}$, $\{v_3, v_4, v_5\}$ and $\{v_6\}$ results in a lesser distance of 12. The following polynomial transformation shows that the bin packing problem also reduces to a special case of the TCVRP. Consider a bin packing problem with items of weights q_i (i=3, ..., n) and bins of size Q. Then construct a star tree as in figure 2.

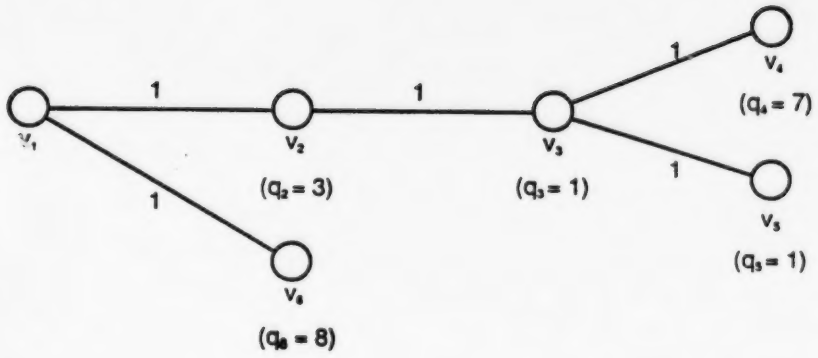


Figure 1: Example showing that minimizing distance does not minimize the number of vehicles

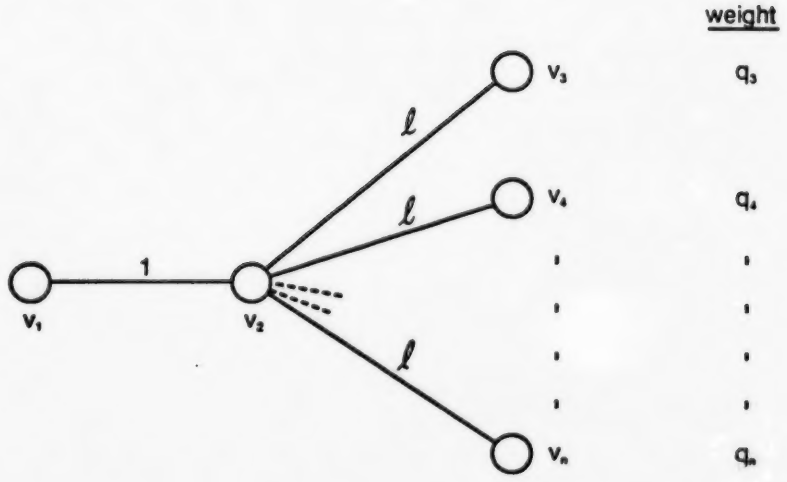


Figure 2: Bin packing transformation into a TCVRP

If $\ell_{1,2}=1$ and $\ell_{2,j}=\ell$ $(j=3, \ldots, n)$, then the optimal TCVRP solution has a value of $z = 2m + 2(n-2)\ell$, where m is the number of vehicles used. Since the second term of z is constant, solving the TCVRP is then equivalent to solving a bin packing problem. The same reduction can be used to show that the TCVRP with splittable demands is also NP-hard. In this case, assume ℓ is sufficiently large (e.g., $\ell > n$) so that, in the optimal solution, each edge (v_2, v_j) $(j=3, \ldots, n)$ is traversed only once in each direction and thus, no split occurs. Then the above argument applies.

The remainder of this paper is organized as follows. In the next two sections, we present lower and upper bounds for the TCVRP, together with worst case behaviours. In section 3, an enumerative algorithm using these bounds is described. Computational results are presented in section 4.

1. Lower bounds

Consider an edge (v_i, v_j) of E (where $t_i < t_j$) and $T_j = (V_j, E_j)$, the maximal subtree rooted at v_j in the opposite direction from the depot. Note that we admit the degenerate case $T_j = (\{v_j\}, \emptyset)$, whenever v_j is a pendant vertex. Let $b(V_j)$ be the optimal value of the bin packing problem associated with the weights of V_j and bins of size Q.

Proposition 1 (lower bound).

A lower bound on z^*, the TCVRP optimal value, is given by

$$\underline{z} = 2 \sum_{(v_i,v_j)\in E} b(V_j)\, \ell_{ij}.$$

Proof.

The result follows immediately from the fact that edge (v_i, v_j) will be traversed at least $2b(V_j)$ times. □

Since the bin packing problem is itself NP-hard, it is convenient to replace $b(V_j)$ by a lower bound $\underline{b}(V_j)$. Several such bounds are available. The simplest one is

$$\underline{b}_1(V_j) = \left\lceil \frac{\sum_{v_k \in V_j} q_k}{Q} \right\rceil .$$

This bound has been improved by Martello and Toth (1988). We present their result with a different notation. Let a and b be such that $0<a<Q$ and $0<b<Q$ and $W_j(a,b) = \{v_k \in V_j : a<q_k<b\}$. (Note that $W_j(a,b)=\emptyset$ wherever $a>b$.) Then the Martello and Toth bound is

$$\underline{b}_2(V_j) = \max_{0<c<Q/2} \{\underline{b}_2(V_j, c)\}$$

where

$$\underline{b}_2(V_j, c) = |W_j(Q/2,Q)| + \max \left\{0, \left\lceil \frac{\sum_{v_k \in W_j(c,Q-c)} q_k - |W_j(Q/2,Q-c)|Q}{Q} \right\rceil \right\} .$$

The computation of this bound rests on the fact that weights larger than $Q/2$ need separate bins and that the weights of $W_j(c, Q/2)$ cannot be accommodated in the bins required for the weights of $W_j(Q-c, Q)$. As shown by Martello and Toth, $\underline{b}_2(V_j)$ can be computed in $O(|V_j|)$ time.

We now propose an improvement to this bound. It uses the fact that no more than two weights of $W_j(Q/3, Q/2)$ can fit in the same bin. To compute this bound, assume first that all weights of V are given in

increasing order so that the ensuing process can still be carried out in $O(|V_j|)$ time once the weights are sorted. As above, the weights of $W_j(Q/2, Q)$ are assigned to separate bins. An attempt is then made to assign the weights of $W_j(Q/3, Q/2)$ to these bins according to an adapted FIRST FIT procedure (Coffman et al. 1984), i.e. by assigning the smallest remaining weight to the remaining bin having the smallest residual capacity. At most one weight of $W_j(Q/3, Q/2)$ can fit in any of these bins. Let K_j be the number of weights of $W_j(Q/3, Q/2)$ which cannot be assigned to a bin through this process. Then at least $\lceil K_j/2 \rceil$ additional bins are required. Hence a valid lower bound is

$$\underline{b}_3(V_j) = \max_{0<c<Q/3} \{\underline{b}_3(V_j, c)\}$$

where

$$\underline{b}_3(V_j, c) = |W_j(Q/2, Q)| + \lceil K_j/2 \rceil + \max\{0, p\}$$

and

$$p = \left\lceil \frac{\sum_{v_k \in W_j(c, Q-c)} q_k - (|W_j(Q/2, Q-c)| + \lceil K_j/2 \rceil)Q}{Q} \right\rceil .$$

Note that values of c can be restricted to the range $[0, Q/3)$ since $p=0$ whenever $c>Q/3$. It is also easy to show that $\underline{b}_3(V_j) = b(V_j)$ whenever $W_j(0, Q/3) = \emptyset$. This process could of course be extended to the weights of $W_j(Q/4, Q/3)$, $W_j(Q/5, Q/4)$, etc., but the resulting marginal improvement in the lower bound would decrease at each step and does not seem justified for moderate values of n and uniform weight distributions. Finally, it is straightforward to establish that $\underline{b}_1(V_j) \leq \underline{b}_2(V_j) \leq \underline{b}_3(V_j)$.

2. Heuristic and upper bound

We now describe a simple linear time heuristic procedure yielding a feasible solution in which each edge (v_i, v_j) is traversed at most $2\,\underline{b}_1(V_j)$ times in each direction. This procedure eliminates at each step a pendant vertex from the current tree and creates a new vehicle route if the sum of the weight acumulated at that vertex and the weight accumulated at its predecessor exceeds Q. Formally, the algorithm can be described as follows. First define:

m : the number of vehicles used in the solution;

$\bar{z}$: the total distance travelled by the m vehicles;

U_k $(k=1, \ldots, m)$: the set of vertices whose weights are collected by the k^{th} vehicle;

S_j $(v_j \in V)$: the set of all vertices whose weights are accumulated at v_j;

d_j $(v_j \in V)$: the total distance accumulated at v_j.

Step 1. Initialization

Set $m:=0$, $\bar{z}:=0$, $d_j:=0$ and $S_j:=\{v_j\}$ $(j=1, \ldots, n)$.

Step 2. Termination check

If $V=\{v_1\}$, set $m:=m+1$, $U_m:=S_1$, $\bar{z}:=\bar{z}+2d_1$ and stop.

Otherwise, select a pendant vertex $v_j \in V \setminus \{v_1\}$. Let v_i be the vertex adjacent to v_j.

Step 3. Route creation from v_j; v_i remains unchanged

If $q_i + q_j < Q$, go to step 5.

Otherwise, if $q_j < q_i$, go to step 4. Set $m := m+1$, $U_m := S_j$, $\bar{z} := \bar{z} + 2d_j + 2t_j$, $V := V \setminus \{v_j\}$, $E := E \setminus \{(v_i, v_j)\}$. Go to step 2.

Step 4. Route creation from v_i; v_i is replaced by v_j

Set $m := m+1$, $U_m := S_i$, $q_i := q_j$, $S_i := S_j$, $\bar{z} := \bar{z} + 2d_i + 2t_i$, $d_i := d_j + \ell_{ij}$, $V := V \setminus \{v_j\}$, $E := E \setminus \{(v_i, v_j)\}$. Go to step 2.

Step 5. Merging v_i and v_j

Set $q_i := q_i + q_j$, $d_i := d_j + \ell_{ij}$, $S_i := S_j \cup S_i$, $V := V \setminus \{v_j\}$, $E := E \setminus \{(v_i, v_j)\}$. Go to step 2. □

The linearity of this procedure follows from the fact that each vertex is considered only once. The worst case behaviour of the heuristic is given by

Proposition 2 (worst case behaviour of the heuristic).

Let $\bar{z}$ be the upper bound provided by the heuristic, on the optimal TCVRP value z^*. Then

$$z^* \leq \bar{z} \leq 4 \sum_{(v_i,v_j)\in E} \underline{b}_1(V_j)\, \ell_{ij}.$$

Proof.

First observe that in steps 3 and 4 of the heuristic, a new vehicle is created only if $q_i + q_j > Q$. In this case, the load of this vehicle from v_i to the depot is equal to Q if $q_i + q_j = Q$ or equal to $\max\{q_i, q_j\}$, otherwise. In either case, this load is strictly larger than $Q/2$. This implies that at most one of all vehicles traversing any edge (v_i, v_j) of E in any given direction has a load not exceeding $Q/2$. This number of vehicles does not exceed $\bar{b}(V_j) = \left\lceil \frac{\sum_{v_k \in V_j} q_k}{Q/2} \right\rceil$, which is never more than twice the lower bound $\underline{b}_1(V_j) = \left\lceil \frac{\sum_{v_k \in V_j} q_k}{Q} \right\rceil$. The conclusion follows immediately from this observation, taking two-directional trips into account. □

Corollary.

The lower bound provided by Proposition 1 is never less than $z^*/2$ and the heuristic is a 2-approximation algorithm.

3. Enumerative algorithm

In this section, we present a breadth-first branch and bound algorithm for the TCVRP. In the sequel, upper indices refer to node numbers of the search tree.

At the first node of the search tree, an initial feasible solution of value $\bar{z}$ is computed by means of the heuristic described in section 2, and a lower bound $\underline{z}$ is computed as in Proposition 1, using $\underline{b}_3(V_j)$ instead of $b(V_j)$. If $\underline{z}=\bar{z}$, the heuristic solution is optimal. Otherwise, the branching process is initiated. All vertices of $W_1(Q/2, Q)$ are first assigned to different vehicles and the set of routed vertices is defined as $I^1 = W_1(Q/2, Q)$.

At any iteration, the pendant node h of the search tree having the least lower bound $\underline{z}^h$ on z^* is selected for branching. The set I^h of already assigned vertices is considered, and a new vertex v_j is removed from $F^h=V\setminus I^h$, the set of free (unassigned) vertices, so that t_j is maximum. Branches are created from h by successively assigning v_j to each of the existing vehicles (when this is feasible) and to a new vehicle. A lower bound $\underline{z}^g$ on z is computed for every potential node g descending immediately from h. Whenever $\underline{z}^g > \bar{z}$, the value of the incumbent, node g is not created. Otherwise, I^g is set equal to $I^h \cup \{v_j\}$. The value of $\bar{z}$ is updated at every node h for which F^h is empty. We now describe the computation of the lower bound $\underline{z}^g$.

Let g be a node of the search tree descending immediately from node h and v_j, the vertex just considered for branching. Define $\underline{b}^g(V_i)$ as a lower bound on $b(V_i)$, given that vertices of $I^g \cap V_i$ have already been assigned to vehicles. As indicated above, $\underline{b}^1(V_i) = \underline{b}_3(V_i)$ for all i. At node g, define $P_j = \{v_k: v_k \text{ is on the path from } v_j \text{ to } v_1\}$. Then $\underline{b}^g(V_i)$ may be different from $\underline{b}^h(V_i)$ only if $v_i \in P_j$. In such a case, it is no longer possible to apply the process used to compute $\underline{b}_2(V_i)$ or $\underline{b}_3(V_i)$ since the weight assignments already made may invalidate the computation of the minimum number of additional vehicles required. Instead, we propose the following procedure. Consider the m vehicles already assigned to vertices of V_i, with residual capacities $r_1, \ldots, r_m$ ranked in non-decreasing order. Let M_1 be the set of vehicles k with $r_k < 2Q/3$ and M_2, the set of the $m-|M_1|$ remaining vehicles. Also define $W_i^g = F^g \cap W_i(Q/3, Q/2)$. Using the adapted FIRST FIT procedure described in section 2, determine the number m_1 of weights of W_i^g which can be assigned to vehicles of M_1. Since these vehicles can each receive at most one weight of W_i^g, and vehicles of M_2 at most two weights of W_i^g, the minimum number of additional vehicles required for the weights of W_i^g is then equal to

$$m' = \max\{0, \lceil (|W_i^g| - m_1 - 2|M_2|)/2 \rceil\}.$$

Now, let m" be the minimum number of additional vehicles required beyond m+m' for the weights of $F^g \cap W_i(0, Q/3)$:

$$m'' = \max\left\{0, \left\lceil \sum_{v_k \in F^g \cap W_i(0, Q/2)} q_k - \sum_{k=1}^{m} r_k - m'Q \right\rceil\right\}$$

and note that

$$\underline{b}^g(V_i) = \max\{\underline{b}_3(V_i), m+m'+m''\}$$

is a valid lower bound on $b(V_i)$. The following proposition shows that there is no need in fact to compute m".

Proposition 3.

$\max\{\underline{b}_3(V_i), m+m'+m"\}$ is always equal to $\max\{\underline{b}_3(V_i), m+m'\}$.

Proof.

If $\underline{b}_1(V_i) < m+m'$, then obviously m"=0; if, on the other hand, $\underline{b}_1(V_i) > m+m'$, then $m+m'+m" = \underline{b}_1(V_i) < \underline{b}_3(V_i)$. □

Using Proposition 1, the lower bound $\underline{z}^g$ on z*, computed at node g of the search tree, is then

$$\underline{z}^g = 2 \sum_{(V_i,V_j)\in E} \underline{b}^g(V_j)\, \ell_{ij}.$$

4. Computational results

The algorithm just described was coded in FORTRAN and tested over a set of random problems generated as follows. For n=20, 40, ..., ten trees were generated each time by creating between 1 to 5 descendant from each node, according to a discrete uniform distribution. For each tree, integer weights q_i were randomly and uniformly generated in each of the intervals [10,90], [20,80], [30,70], [1,100], [1,50], [1,30], [1,10], [10,10], [20,20] and [30,30]. Edge lengths were generated according to a discrete uniform distribution on [1,100]. The vehicle capacity was always equal to 100. For each problem, the maximum number of vehicles was determined by means of the heuristic described in section 2.

All problems were solved on the University of Montreal CYBER 855 Computer, but all reported times are CYBER 173 equivalent. A time limit of 100 seconds was set on the solution of any problem. The results reported in table 1 are average values on the 10 problems which were attempted and could be solved within the prescribed time limit. The column headings are as follows:

n : number of vertices;

[a,b] : interval within which the weights were generated;

SUCC : number of problems (out of 10) which could be solved within 100 seconds;

NODES : total number of nodes created in the search tree;

BRANCH: number of these nodes which were considered for branching;

HEUR : $\bar{z}/z^*$;

ROOT : $\underline{z}^1/z^*$;

TIME : number of CPU seconds.

Our computational results indicate that the algorithm was successful in solving problems involving up to 140 vertices. Larger problems with up to 200 vertices were solved, but less consistently. Weight distribution appears to be a major factor affecting problem difficulty. In particular, problems with weights in [1,b] become easier as b decreases. Problems with identical weights, i.e. those generated in intervals [a,a] are also easier for small values of a. On successful problems, the solution value provided by the heuristic is in most cases within 5% of the optimum and the value of the lower bound at the root of the search tree is almost always superior to 97% of the optimum. These results compare favourably with those obtained for problems generated over general graphs, where optimal solutions for instances in excess of 50 vertices are rare (Laporte and Nobert 1987).

Acknowledgements

This work was partially supported by the Canadian Natural Sciences and Engineering Research Council, grant A4747. Thanks are also due to Matteo Fischetti who suggested Proposition 3.

Table 1 : Computational results

n	[a,b]	SUCC	NODES	BRANCH	HEUR	ROOT	TIME
20	[10,90]	10	67.9	23.7	1.05	0.97	0.52
	[20,80]	10	56.8	18.6	1.06	0.97	0.41
	[30,70]	10	84.2	20.6	1.06	0.98	0.58
	[1,100]	10	64.3	47.7	1.03	0.98	0.66
	[1,50]	10	302.3	158.4	1.03	1.00	2.65
	[1,30]	10	1.0	0.0	1.00	1.00	0.04
	[1,10]	10	1.0	0.0	1.00	1.00	0.04
	[10,10]	10	1.0	0.0	1.00	1.00	0.04
	[20,20]	10	4.8	2.0	1.00	1.00	0.07
	[30,30]	10	1,114.4	918.2	1.04	0.99	12.63
40	[10,90]	8	1,193.0	135.0	1.06	0.97	12.75
	[20,80]	5	2,127.6	299.2	1.07	0.97	32.05
	[30,70]	7	2,318.3	318.8	1.06	0.96	38.58
	[1,100]	9	1,935.4	222.2	1.07	0.97	32.47
	[1,50]	7	445.8	85.0	1.04	1.00	4.17
	[1,30]	10	1,692.8	685.9	1.03	1.00	31.85
	[1,10]	10	1.0	0.0	1.00	1.00	0.10
	[10,10]	10	12.6	7.0	1.00	0.99	1.00
	[20,20]	10	63.1	31.6	1.01	1.00	1.10
	[30,30]	4	639.0	343.0	1.04	1.00	2.75
60	[30,70]	2	2,723.0	172.5	1.11	0.89	44.67
	[1,100]	1	2,185.0	155.0	1.08	0.92	27.89
	[1,50]	1	1,358.0	187.0	1.17	0.83	15.86
	[1,30]	9	1,400.2	294.6	1.03	0.97	32.50
	[1,10]	10	27.2	22.1	1.00	0.99	1.51
	[10,10]	10	515.4	290.1	1.02	1.00	19.09
	[20,20]	8	748.0	102.5	1.04	1.00	10.05
80	[1,10]	9	11.8	8.7	1.00	0.99	1.09
	[10,10]	6	373.8	126.1	1.01	1.00	17.10
	[20,20]	8	1,547.8	153.6	1.04	1.00	31.45
100	[1,30]	1	623.0	102.0	1.01	1.00	20.74
	[1,10]	9	328.1	155.1	1.01	1.00	23.64
	[10,10]	3	3,210.3	623.3	1.02	1.00	91.10
	[20,20]	3	3,651.0	400.3	1.05	1.00	87.70
120	[1,10]	8	345.8	124.7	1.01	1.00	26.65
140	[1,10]	10	92.4	50.8	1.00	0.99	14.75

References

Altinkemer, K. and B. Gavish. 1985. Heuristics for Equal Weight Delivery Problems with Fixed Error Guarantee. Working Paper QM8536, Graduate School of Management, University of Rochester, NY.

Altinkemer, K. and B. Gavish. 1987. Heuristics for Unequal Weight Delivery Problems with Fixed Error Guarantee. Operations Research Letters 6, 149-158.

Christofides, N. 1989. Worst Case Analysis of a New Heuristic for the Traveling Salesman Problem. Mathematical Programming (to appear).

Christofides, N. 1985. Vehicle Routing. In The Traveling Salesman Problem, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan and D.B. Shmoys, eds., Wiley, Chichester, 431-448.

Coffman, E.L., Garey, M.R. and D.S. Johnson. 1984. Approximation Algorithms for Bin-Packing – An Updated Survey. In Algorithm Design for Computer System Design, G. Ausiello, M. Lucertini and P. Serafini, eds., Springer, Vienna, 49-106.

Golden, B.L. and A.A. Assad. 1988. Vehicle Routing: Methods and Studies, North-Holland, Amsterdam.

Haimovich, M., Rinnooy Kan, A.H.G. and L. Stougie. 1988. Analysis of Heuristics for the Vehicle Routing Problem. In Vehicle Routing: Methods and Studies, B.L. Golden and A.A. Assad, eds., North-Holland, Amsterdam, 47-62.

Laporte, G. and Y. Nobert. 1987. Exact Algorithms for the Vehicle Routing Problem. In Surveys in Combinatorial Optimization, S. Martello, G. Laporte, M. Minoux and C. Ribeiro, eds., North-Holland, Amsterdam, 147-184.

Martello, S. and P. Toth. 1988. Lower Bounds and Reduction Procedures for the Bin Packing Problem. Research report, DEIS, University of Bologna.